Basu 1-1

Official 7/5/03

REMARKS

Claims 1-6, 12-14, and 24-25 were rejected under 35 USC 103 as being unpatentable over Bak, US Patent 6,212,608, in view of Brandle et al US Patent 5,218,699. Applicants respectfully traverse.

Bak deals with the issue of locking and unlocking of software objects in an object-based system. A software object is a , construct that has a state, and a set of operations through which the state may be altered by the application of an input. In a multi-threaded system, where each thread can independently access an object and proceed to modify its state, it is obviously important to prevent a thread from modifying an object that is in the midst of being modified by another thread. To prevent this, the notion of locking an object has been introduced in the art, whereby a thread is not allowed to access an object in a manner that permits modifications, unless the object is unlocked. When the object is found to be unlocked, the thread locks the object and then can operate on it.

In the prior art described by Bak, in order to make certain that only the thread that possesses an object lock is allowed to operate on the locked object, an associated synchronization construct is accessed by the thread to determine whether the object is locked or unlocked. A synchronization construct is a set of data structures that implement a dynamic association between the synchronization construct and an object. The synchronization construct of Bak includes a counter, an object pointer that identifies the associated object, and an identifier for the thread that currently has locked the synchronization construct and thus locked the associated object.

The problem with the above approach, according to Bak, is that these constructs require a significant amount of memory (on the order of the memory occupied by the objects). Additionally, since a particular construct is associated with an object through a pointer that is included in the construct, when a thread desires to access an object for modification purposes it needs to search through the set of stored constructs to locate the construct that associated with the particular object to which access is desired in order to determine whether the object is locked, and that consumes processing time. In short, the use of separate constructs to tract the status of objects consumes significant resources and

is, therefore, a relatively inefficient approach. Bak set out to find a more efficient approach to the problem of locking objects and identifying whether an object is locked.

In the approach described by Bak, the locking status of an object is implicitly recorded in the header of the object, which generally contains information about the object. When a thread – which is a synchronized method that is executed via a stack – wishes to lock an unlocked object, it places a reference value that identifies the thread in the objects header. When another thread wishes to lock the object, a simple reference to the object's header reveals that the header is locked, and also identifies the thread that locked the object.

The Examiner quotes the preamble and the first clause of applicants' claim 1, and asserts that Bak discloses the quoted language. Applicant respectfully disagrees.

A multi-threaded system does have a plurality of stacks and, as pointed out by the Examiner, each thread has its own execution stack. Each such stack contains one or more entries that change with time as the execution of a thread proceeds. Thus, at one point in time a stack may store the header value 314 (see col. 7, lines 29-30), and at another time the same stack may contain some other value.

In contradistinction, amended claim 1 specifies an arrangement wherein each stack in a plurality of stacks includes "a **predefined set** of at least one mediation module that processes an applied signal to form a signal that is applied to said at least one resource of said collection of resources" (emphasis supplied). This is totally different from the kind of information that is contained in the Bak stacks.

The Examiner is not using the Bak reference alone. However, the second reference (Brandle et al) that is cited is applied only to the second clause of claim 1. Therefore, even if the second reference were to constitute a perfect correspondence to the second clause of claim 1, the combination of the Bak and Brandle et al references would still not suggest the kind of stacks (i.e., each containing a predefined set of one or more mediation modules) that amended claim 1 specifies and, therefore, applicants respectfully submit that amended claim 1 is patentable over the references by virtue of the definition contained in the first clause.

Admitting that Bak does not specifically disclose a service director as called for by the second clause of claim 1, the Examiner proceeds to quote the second clause and

assert that Brandle et al describe that which the quoted second clause specifies. Applicant respectfully disagrees.

Brandle et al describe an arrangement for routing procedure calls in a network environment comprising a plurality of interconnected computers, each of which can be making procedure calls. Each procedure call made by a computer is routed through a service director module (12) within the computer, and the service director determines whether the procedure can be executed by the service director. If the service director determines that it cannot execute the requested procedure, it then determines whether the procedure is stored within (and therefore can be executed by) the computer, and if not, whether another computer on the network does store the requested procedure. Whatever the case, the service director conditions the procedure call based on the requirements of the module that will execute the procedure call (itself, a separate procedure within its own computer, or a separate procedure within a remote computer), and proceeds to forward the procedure to that module.

Based on the above, applicants respectfully submit that the teachings of Brandle et al do not correspond to the second clause of amended claim 1. First, it is noted that while the service director of Brandle et al accepts procedure calls, it does not classify the calls into "types" of requests because they are all one type: procedure calls. In contradistinction, the second clause of amended claim 1 specifies that the service director "classifies said requests in accordance with said types of said requests." Second, even if the Examiner were to argue that the "classifying" in Brandle et al is the classifying relative to where a procedure is to be executed, it still remains that the Brandle et al service director directs the procedure calls to distinct modules (itself, a procedure stored within its computer, or a procedure stored in a remote computer). These are not processing stacks. In contradistinction, the second clause of amended claim 1 specifies that the service director "directs said requests to different ones of said processing stacks." Third, the second clause of claim 2 specifies a rigid correspondence between the type of request and where it is directed. Specifically, the second clause of claim 2 specifies that each request of a particular type is directed to a stack that is devoted to that type. That is totally different from the kinds of values that are found in the stacks of Bak. Consequently, it is respectfully submitted that the differences in each of the amended

claim 1 clauses and the respective references (Bak for the first clause, and Brandle et al for the second clause) are such that each of the clauses defines subject matter that renders amended claim 1 not obvious in light of the Bak and Brandle et al combination of references.

Additionally, applicants respectfully submit that there is no motivation to combine the two references. As indicated above, Bak deals with how to have one process lock an object so that another process does not concurrently alter it. There is no need for a service director, and including one would not improve the Bak system and, therefore, there is no motivation for doing so.

If one were to combine the two references, the system that would result would be multi-computer network where each computer is a multi-threaded system, and where each thread has its own stack. At any time, each stack may have one or more pointers to procedures that are not predetermined but dependent on the particular program that is being executed, the procedures may have references to objects, and the header fields of those objects would indicate whether the objects are locked or not. When a procedure is popped from a stack of a computer, a service director would decide which module executes the popped procedure. Clearly, that does not correspond to the system defined in amended claim 1, where each stack corresponds to a type of resource request, where the modules within a stack a predetermined, and where the service director directs the stack to which a resource request is sent.

Claims 2-6, 12-14, and 24-25 depend on amended claim 1 and, therefore, are believed to also not be obvious in view of the Bak and Brandle et al combination of references.

Claims 7-8, 10-11, 15, and 23 were rejected under 35 USC 103 as being unpatentable over Bak in view of Brandle et al and further in view of Hershey et al, US Patent 5,414,833. Applicants respectfully traverse.

Hershey et al describe a network security system where a set of parallel finite state machines that are responsive to the bit stream on a selected bus form a security monitor. The Examiner cited col. 23, lines 33-65 where it is described that the security monitor can be adapted to detect whether the bit stream represents encrypted data or plaintext data.

Basu 1-1

Claim 7, being dependent on amended claim 1 is believed allowable by virtue of the fact that it is dependent on amended claim 1 and the Hershey et al reference does not provide that which the combination of Bak and Brandle et al are missing relative to amended claim 1 (and the Examiner does not assert that it does). Additionally, claim 7 specifies that at least one of the mediation modules that is stored in one of the stacks (that contains a predefined set of one or more mediation modules) is based upon a chosen security policy. The monitor of Hershey et al may be based on a chosen security policy, so it is assumed that the Examiner is asserting that the monitor of Hershey et al corresponds to the "at least one mediation module" that is defined in claim 7. However, the monitor of Hershey et al it is not a module in a stack. Indeed, it appears that the monitor of Hershey et al is not a software module but, rather, a set of parallel hardware finite state machines. Whatever is stored in the stacks of the Bak system, or in the stacks of the Bak system modified by Brandle et al, is transitory elements that are related to whatever program a user, or a plurality of users, wish to execute. These transitory elements are not, and cannot correspond to, a parallel set of finite state machines that Hershey et al described.

The rejection of the other claims is also traversed based on the inherently different structure that amended claim 1 imposes. Thus, relative to claim 8 for example, it is true that Hershey et al assume that encryption is performed in the application layer, but claim 8 specifies that it is one of the modules that is in the stack performs an encryption, and there is no teaching in Hershey et al that the application layer employs any stack in performing its encryption. Moreover, the Examiner effectively asserted in connection with claim 7 a correspondence between the *monitor* of Hershey et al and the modules in the stack, but the encryption in the application layer is not the monitor.

Similarly in connection with claim 10, the Examiner points to the monitor of Hershey et al that is constructing a "security alert message." However, the monitor is not in the stack. In connection with claim 11, the Examiner takes notice that secure file systems are known. However, the thrust of claim 11 is that the secure file system is implemented via a mediation module in a particular stack of a set of stacks, as specified in amended claim 1. Therefore, the mere knowledge that secure file systems can be created adds little to the combination of Bak, Brandle et al and Hershey et al.

Basu 1-1

Accordingly, it is believed that claim 11 is not obvious in light of these references and the official notice.

The rejection of claim 15 in this grouping is somewhat puzzling since claim 15 depends on claim 14, and claim 14 is not rejected in this grouping. However, applicants respectfully traverse the rejection of claim 15. The Examiner asserts that in a system that includes a secure file system "it follows that the system would produce a secure connection." Respectfully, that is not the case. The file system may be a secure file system, but the signal applied to the secure file system may be compromised, if for example the network connection to the system (see applicants' claim 14) is not secure.

Claim 23 specifies a module that retrieves an authentication code, and a module that validates a service request against the authentication code. The Examiner basically points to a means within the monitor of Hershey et al ("security alert message transmission means 306") that causes the security alert message and message authentication code to be transmitted. Respectfully, that does not meet the limitation of claim 23. Means 306 is not a module on a stack, there is no teaching that it **retrieves** an authentication code, and means 306 does not serve as a module that validates a service request against the authentication code. Hence, applicants believe that claim 23 is clearly not obvious in view of the cited references.

Claim 9 was rejected under 35 USC 103 as being unpatentable over Bak in view of Brandle et al, and further in view of Traversat et al, US Patent 6,052,720. Applicants respectfully traverse. The Examiner asserts that Traversat et al describe a namespace manager that controls how the entries are stored and accessed. That may be the case, but the Traversat et al reference does not describe a namespace manager that is embodied in a module, which module is in a stack that includes a set of predefined modules. Therefore, it is respectfully submitted, that claim 9 is not obvious in view of the Bak – Brandle et al – Traversat et al combination of references.

Claim 16 was rejected under 35 USC 103 as being unpatentable over Bak in view of Brandle et al, and further in view of Pierce, Jr. et al, US Patent 6,560,217. Applicants respectfully traverse. Claim 16 specifies the system of claim 14, where the network is a virtual private network. The Examiner asserts that Pierce, Jr. et al describe a virtual private network. That may be so, but the Bak reference deals with a single computer, and

Basu 1-1

the Brandle et al reference deals with computers that are closely connected (allowing procedure calls to be executed on different computers of the network). Such an arrangement, most likely, contemplates a local area network. It would not be obvious that a virtual private network – which is an overlay over the public network and contemplates communication between geographically distant machines – would be used to execute different procedures of a given program, because the delays associated with the communication over the virtual private network would make the arrangement work poorly. Therefore, it is respectfully submitted that claim 16 is not obvious over the cited prior art by virtue of its dependence on amended claim 1 and by virtue of the fact that there is no motivation for, and an actual disadvantage in, the notion of adopting the teachings of Pierce, Jr. to the Bak and Brandle et al combination of references.

Claims 17 and 18 were rejected under 35 USC 103 as being unpatentable over Bak in view of Brandle et al, Pierce, Jr. et al, and Hershey et al. Applicants respectfully traverse, for the reasons expressed above in connection with, inter alia, claims 8 and 15.

Claims 19-22 were rejected under 35 USC 103 as being unpatentable over Bak in view of Brandle et al, and further in view of Bond, US Patent 6,275,938. Applicants respectfully traverse.

Claim 19 specifies a system that includes a compliance supervisor that is coupled to both the processing stacks and the service direction, which are particularly defined in claim 1. The Examiner asserts that Bond et al disclose a compliance supervisor, citing col. 8, lines 12-27. What the Examiner is pointing to is a module (WHKRNL32) that implements the security policy. Even if module WHKRNL32 does constitute a "compliance supervisor", it still remains that claim 19 specifies that the compliance supervisor is one that is adapted for receiving security information from outside the system, and there is no indication in Bond et al that module WHKRNL32 is modifiable. If it is not modifiable, then it is NOT adapted "for receiving security policy information from outside said system," as claim 19 specifies.

Additionally, module WHKRNL32 is not connected to any stacks, is not connected to a service director, and is certainly not connected to both, as claim 19 specifies.

Basu 1-1

Actually, the implementation of the security policy, which as the Examiner has pointed out is carried out by module WHKRNL32, is not carried by the compliance supervisor but, rather, by the mediation modules. Thus there is a complete lack of correspondence between, the teachings of Bond et al and the compliance supervisor of claim 19. Accordingly, it is respectfully submitted that claim 19, and claims 20-22, which depend on claim 19, are not obvious in view of the Bak, Brandle et al and Bond et al combination of references.

Claims 26-34 were rejected under 35 USC 103 as being unpatentable over Bak in view of Brandle et al and further in view of Eliott, US Patent 6,468,160. Applicants respectfully traverse, for the reasons expressed in connection with claim 1.

In light of the above amendments and remarks, applicants respectfully submit that all of the Examiner's rejections have been overcome. Reconsideration and allowance are respectfully solicited.

Respectfully,
Anindya Basu
Suvo Mittra

Dated: 7/5/03

By _____
Henry T. Brendzel
Reg. No. 26,844
Phone (973) 467-2025
Fax (973) 467-6589
email brendzel@comcast.net